**Things You Don’t Know About Next.js**

[[](https://medium.com/@herocodes?source=post_page-----02ee54cb5b7f--------------------------------)](https://medium.com/@herocodes?source=post_page-----02ee54cb5b7f--------------------------------)

[[JavaScript in Plain English](https://javascript.plainenglish.io/?source=post_page-----02ee54cb5b7f--------------------------------)](https://javascript.plainenglish.io/?source=post_page-----02ee54cb5b7f--------------------------------)

[Hero](https://medium.com/@herocodes?source=post_page-----02ee54cb5b7f--------------------------------)

·

Follow

Published in

[JavaScript in Plain English](https://javascript.plainenglish.io/?source=post_page-----02ee54cb5b7f--------------------------------)

·

6 min read

·

5 days ago

349

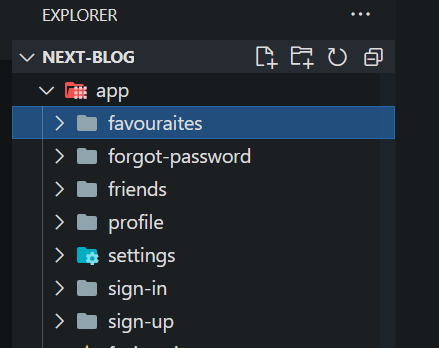
NextJS 14 isn’t just an update, it’s a refocus on developer experience and speed. Overall, NextJS 14 promises a faster, more enjoyable development journey for React developers. Whether you’re starting fresh or upgrading an existing project, it’s definitely worth checking out.

Today I am going to tell you about some concepts of NextJS that most of the developers don’t know, you can use them to optimize your App and improve the Developer Experience.



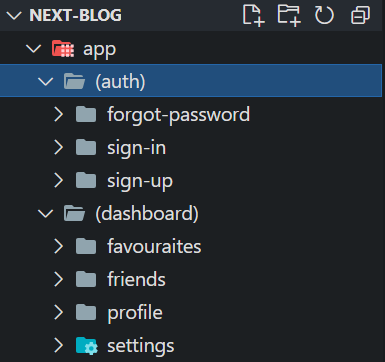
So let’s begin 😉

[***1. Route Groups***](https://nextjs.org/docs/app/building-your-application/routing/route-groups)



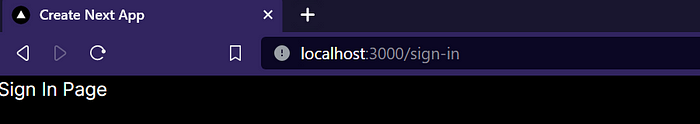
Messy Folder Structure

Do you have a messy folder structure? Hard to find the particular route? So organize them by using Route Group functionality provided by Next JS.  
Let’s organize this folder structure using Route Groups.



Organized Folder Structure

Now you can easily find routes about different topics by keeping them under the Route Group Folder.  
**Route groups don’t add their folder name in the URL**URL: http:localhost:3000/sign-in



No (auth) in the URL.

[***2. Static Metadata***](https://nextjs.org/docs/app/building-your-application/optimizing/metadata#static-metadata)

Next.js has a Metadata API that can be used to define your application metadata (e.g. meta and link tags inside your HTML head element) for improved SEO and web shareability.

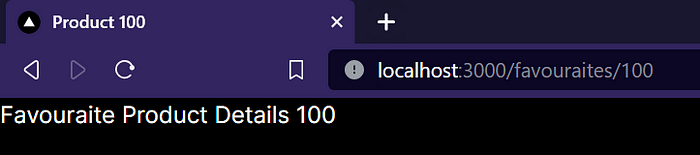
You can use the metadata API in both page.tsx or layout.tsx

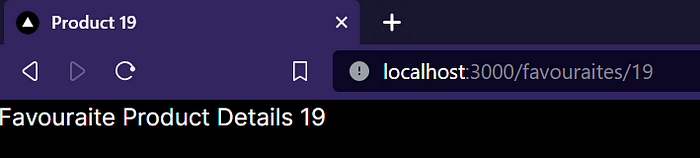
import type { Metadata } from 'next'  
   
export const metadata: Metadata = {  
 title: 'Hero's Blog',  
 description: 'Blog created by Hero',  
}

[***3. Dynamic Metadata***](https://nextjs.org/docs/app/building-your-application/optimizing/metadata#dynamic-metadata)

You can use generateMetadata function to fetch metadata that requires dynamic values.  
It is used to increase and enhance the SEO score of your website.

import type { Metadata } from "next";  
  
type Props = {  
 params: {  
 id: string  
 }  
};  
  
export const generateMetadata = ({ params }: Props): Metadata => {  
 return {  
 title: `Product ${params.id}`  
 }  
}  
  
export default function FavouriteProductDetails({ params }: Props) {  
 return <h1>Favouraite Product Details {params.id}</h1>  
}





As you can see above, dynamically the title of the website is getting changed.

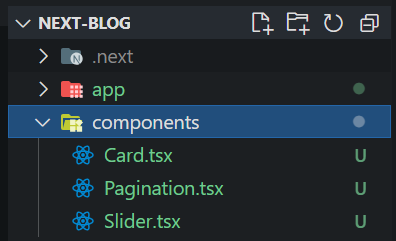
[**Ordering of Metadata**](https://nextjs.org/docs/app/building-your-application/optimizing/metadata#ordering)Metadata is evaluated in order, starting from the root segment down to the segment closest to the final page.tsx segment. For example:

1. app/layout.tsx (Root Layout)
2. app/favourite/layout.tsx (Nested Blog Layout)
3. app/favourite/[slug]/page.tsx (Blog Page)

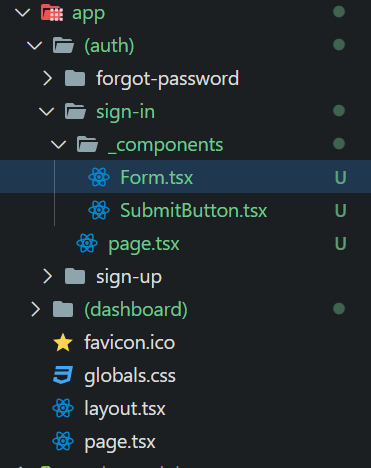
***4. Private Routes***

You might be thinking what are private routes? Are they something which only an admin can access? No, private routes means the folders which can’t be accessed by any user directly through the website. Simply the web pages that aren’t served directly to the client.  
This can be achieved by the following methods:

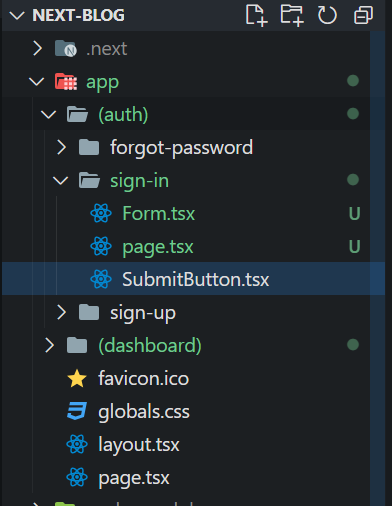
1. Making a separate components directory outside the app directory.



2. Inside any directory, which is under the app directory, creating a \_components folder. (Any name can be given, and yeah that’s an underscore, you saw that right)



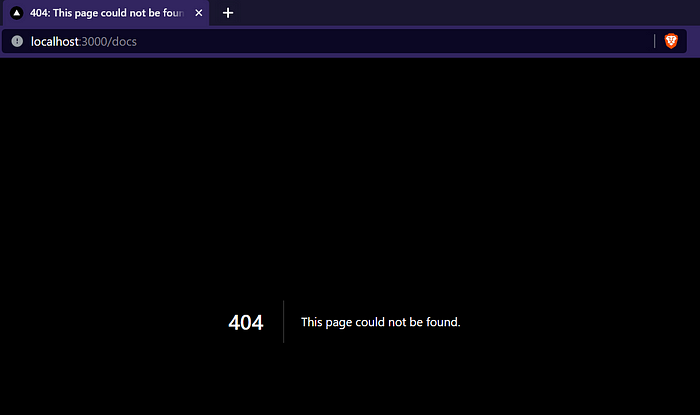
3. Creating different files inside a directory which won’t get served directly to the client until and unless added to the page.tsx file.



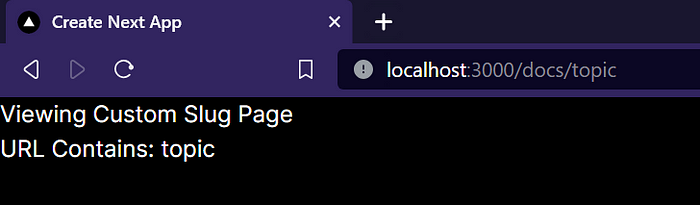
[***5. Catch-all Segments***](https://nextjs.org/docs/pages/building-your-application/routing/dynamic-routes#catch-all-segments)

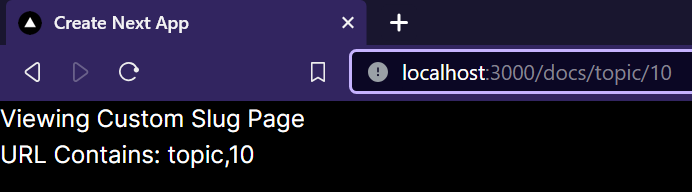
Dynamic Segments can be extended to catch-all subsequent segments by adding an ellipsis inside the brackets [...segmentName]

For example, /docs/[...slug]/page.tsx will match /docs/topic , but also /docs/topic/1 and so on. But if will give us a page not found error if the URL is /docs .



404 error





Here is the code snippet: app/docs/[...slug]/page.tsx

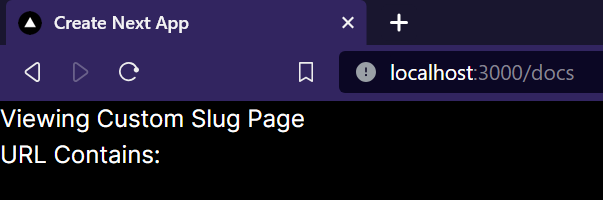
import React from 'react'  
  
type Params = {  
 params: {  
 slug: string[]  
 }  
}  
  
export default function SlugPage({ params: { slug } }: Params) {  
 return (  
 <div>  
 <h1>Viewing Custom Slug Page</h1>  
 <span>URL Contains: {slug.toString()} </span>  
 </div>  
 )  
}

You guys might be wondering can we fix that 404 error? 🤔  
What if I say yes!! Yes you can fix that error.  
Scroll down to see the solution !!

[***6. Optional Catch-All Segments***](https://nextjs.org/docs/pages/building-your-application/routing/dynamic-routes#optional-catch-all-segments)

Catch-all Segments can be made optional by including the parameter in double square brackets: [[...segmentName]].

For example, /docs/[[...slug]]/page.tsxwill also match /docs, in addition to /docs/topic, /docs/topic/10.



No more error!

The difference between **catch-all** and **optional catch-all** segments is that with optional, the route without the parameter is also matched (/docs in the example above).

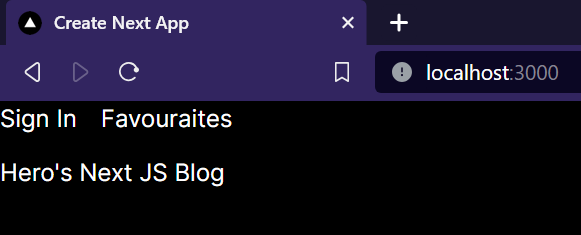
***7. Active Links***

Ever thought how there is an overlay on the links of a site the one which you are viewing on your screen?  
Today I am going to tell you how to achieve that functionality and enhance the User Experience.

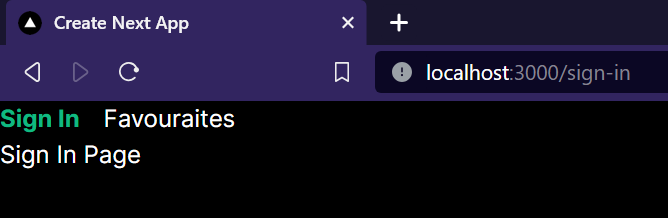
So let us first create a Navbar.tsx in the components directory.  
As this will be a client component because the user will interact with the Navbar, add the "use client" directive at the top your file. Also import a hook name usePathname from next/navigation .

"use client";  
import Link from 'next/link';  
import { usePathname } from 'next/navigation';  
import React from 'react';  
  
type Links = {  
 title: string,  
 url: string  
}  
  
export default function Navbar() {  
  
 const links: Links[] = [  
 {  
 title: "Sign In",  
 url: "/sign-in"  
 },  
 {  
 title: "Favourite",  
 url: "/favourite/1"  
 },  
 ];  
  
 const pathname = usePathname()  
  
 return (  
 <div className='flex space-x-4'>  
 {  
 links.map(({ title, url }: Links) => {  
  
 const isActive = pathname.startsWith(url)  
  
 return <Link  
 className={isActive ? "font-bold text-emerald-500" : "text-white"}  
 key={title}  
 href={url}>  
 {title}  
 </Link>  
 })  
 }  
 </div>  
 )  
}

So let me break the code for you, first, by creating a links constant we are defining the links we want in our Navbar. Map them and your page should somewhat look like this:



Click on any one of the links to see the magic!



As you can see, as I clicked on the “Sign In” link the color of the text got changed! That was so easy :)  
**Note: Make sure the route URL gonna use already exists or it will give a 404 error.**

Follow me on medium for more such superb blogs :) and also don’t forget to follow [JavaScript in Plain English](https://javascript.plainenglish.io/)

Any doubts?  
Contact me:  
Discord: hero.xyz  
[Twitter](https://twitter.com/HeroCodes404)

I hope you’ve enjoyed reading it.

Will be back soon with another blog, to get updated with my blogs follow me on medium or subscribe to the newsletter.

Till that time, have a nice day ahead, and always keep smiling. :)

Thank You,

Hero

**In Plain English 🚀**

*Thank you for being a part of the*[***In Plain English***](https://plainenglish.io/)*community! Before you go:*

* Be sure to **clap** and **follow** the writer ️👏**️️**
* Follow us: [**X**](https://twitter.com/inPlainEngHQ)**|**[**LinkedIn**](https://www.linkedin.com/company/inplainenglish/)**|**[**YouTube**](https://www.youtube.com/channel/UCtipWUghju290NWcn8jhyAw)**|**[**Discord**](https://discord.gg/in-plain-english-709094664682340443)**|**[**Newsletter**](https://newsletter.plainenglish.io/)
* Visit our other platforms: **[Stackademic](https://stackademic.com/" \t "_blank) | [CoFeed](https://cofeed.app/" \t "_blank) |**[**Venture**](https://venturemagazine.net/)
* More content at [**PlainEnglish.io**](https://plainenglish.io/)

[Nextjs](https://medium.com/tag/nextjs?source=post_page-----02ee54cb5b7f---------------nextjs-----------------)

[Typescript](https://medium.com/tag/typescript?source=post_page-----02ee54cb5b7f---------------typescript-----------------)

[JavaScript](https://medium.com/tag/javascript?source=post_page-----02ee54cb5b7f---------------javascript-----------------)

[Web Development](https://medium.com/tag/web-development?source=post_page-----02ee54cb5b7f---------------web_development-----------------)

[Coding](https://medium.com/tag/coding?source=post_page-----02ee54cb5b7f---------------coding-----------------)